ASSIGNMENT-3.3

2303A51295

BATCH-10

TASK-1:

PROMPT:

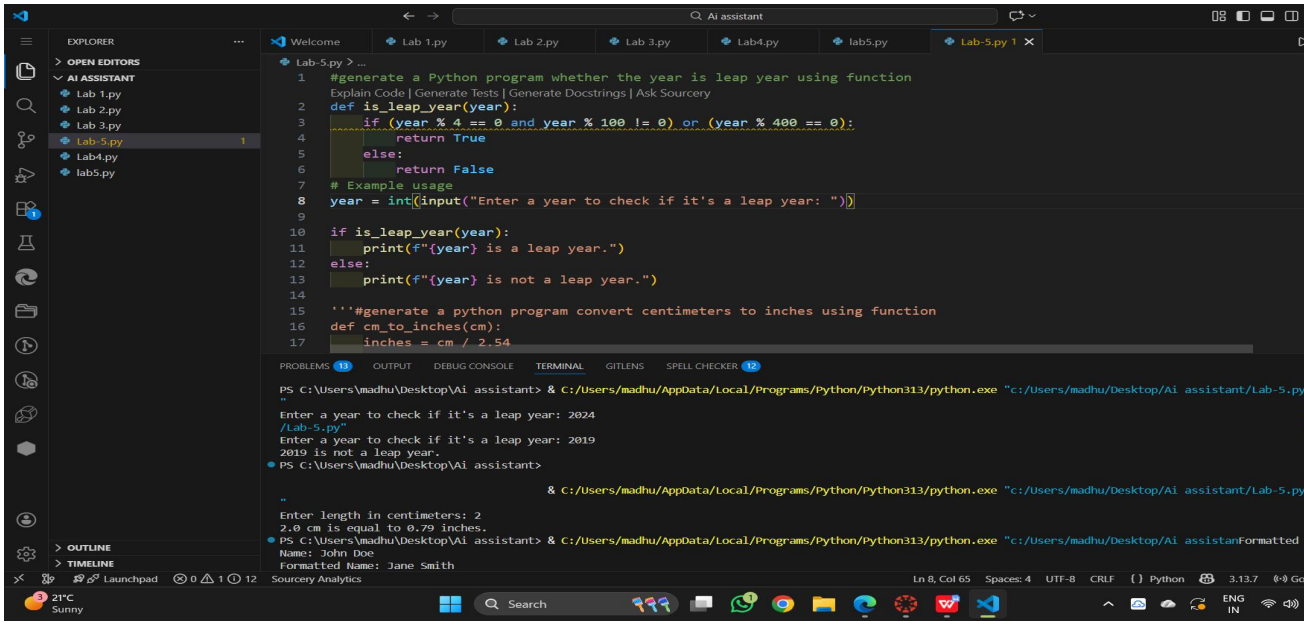
generate a Python program whether the year is leap year using function

CODE:

```
def is_leap_year(year):
    if (year % 4 == 0 and year % 100 != 0) or (year %
400 == 0):
        return True
    else:
        return False
# Example usage
year = int(input("Enter a year to check if it's a leap
year: "))

if is_leap_year(year):
    print(f"{year} is a leap year.")
else:
    print(f"{year} is not a leap year.")
```

OUTPUT:



EXPLANATION:

- This program determines whether a given year is a leap year using a function.

- The function applies standard leap year rules and returns True or False.
- The user inputs a year, which is checked by the function
- The result is printed as either a leap year or not.

TASK-2

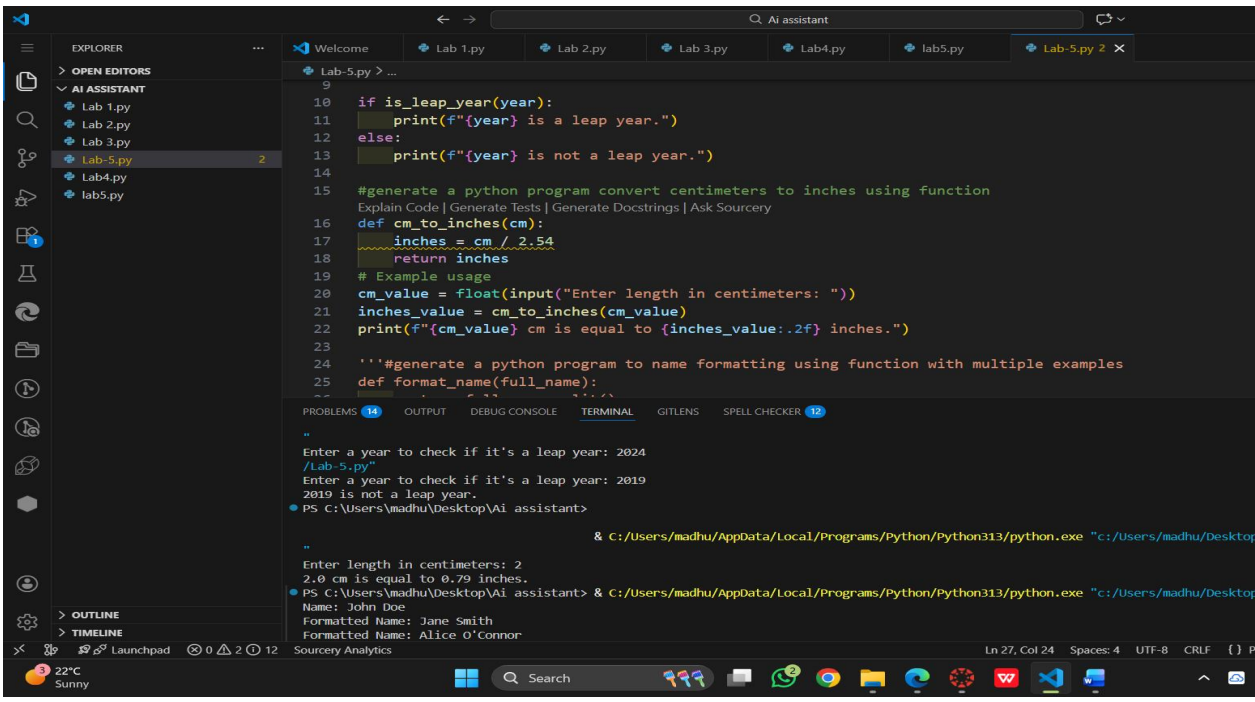
PROMPT:

generate a python program convert centimeters to inches using function

CODE:

```
def cm_to_inches(cm):
    inches = cm / 2.54
    return inches
# Example usage
cm_value = float(input("Enter length in centimeters: "))
inches_value = cm_to_inches(cm_value)
print(f"{cm_value} cm is equal to {inches_value:.2f} inches.")
```

OUTPUT:



EXPLANATION:

- This program converts a length from centimeters to inches using the correct mathematical formula.
- A function performs the conversion by dividing the value by 2.54.
- The user enters a value in centimeters, which is passed to the function.
- The converted result is displayed in inches.

TASK-3:

PROMPT:

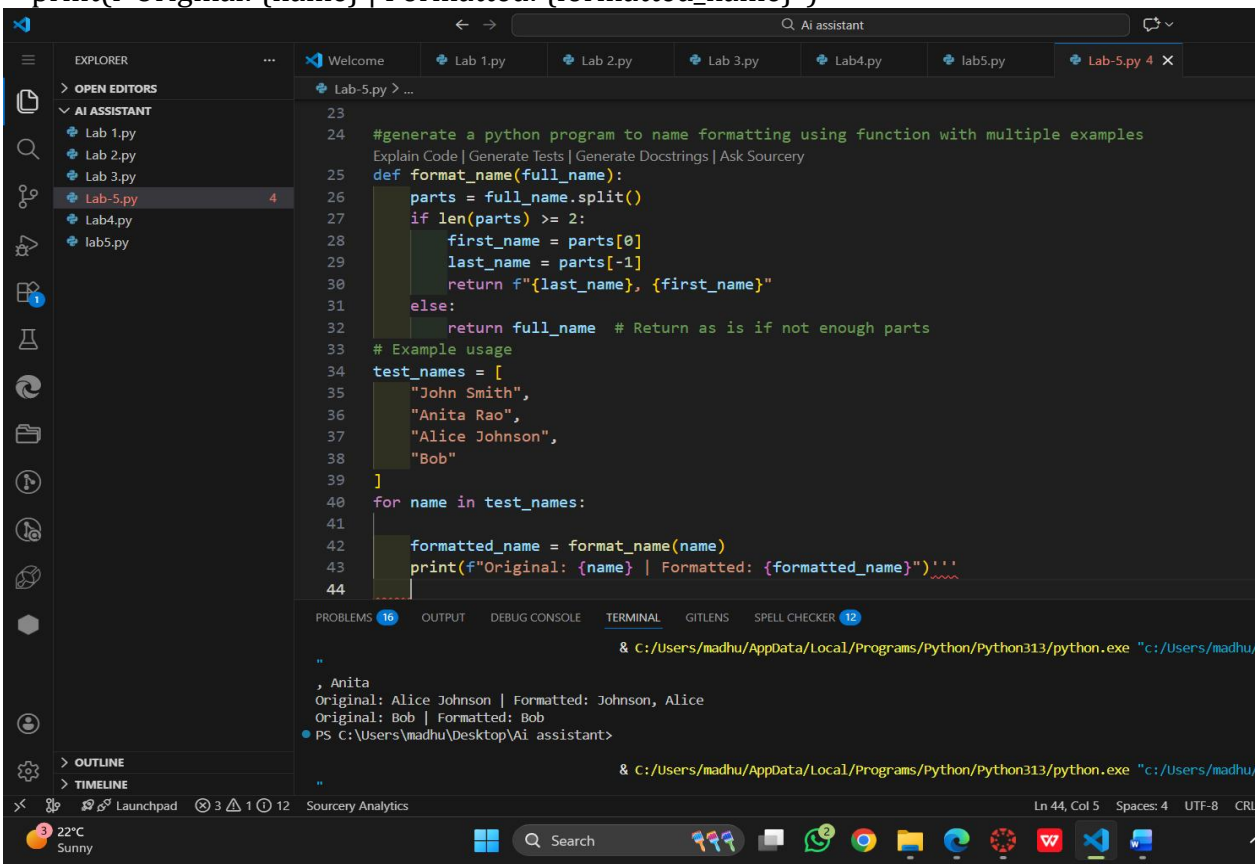
#generate a python program to name formatting using function with multiple examples

CODE:

```
def format_name(full_name):
    parts = full_name.split()
    if len(parts) >= 2:
        first_name = parts[0]
        last_name = parts[-1]
        return f"{last_name}, {first_name}"
    else:
        return full_name  # Return as is if not enough parts
# Example usage
test_names = [
    "John Smith",
    "Anita Rao",
    "Alice Johnson",
    "Bob"
```

]
for name in test_names:

formatted_name = format_name(name)
print(f"Original: {name} | Formatted: {formatted_name}")



OUTPUT:

EXPLANATION:

- This program formats a full name entered as first name and last name.
- The input is split and validated to ensure the correct format.
- Each part of the name is cleaned and capitalized.
- The formatted full name is then displayed.

TASK-4:

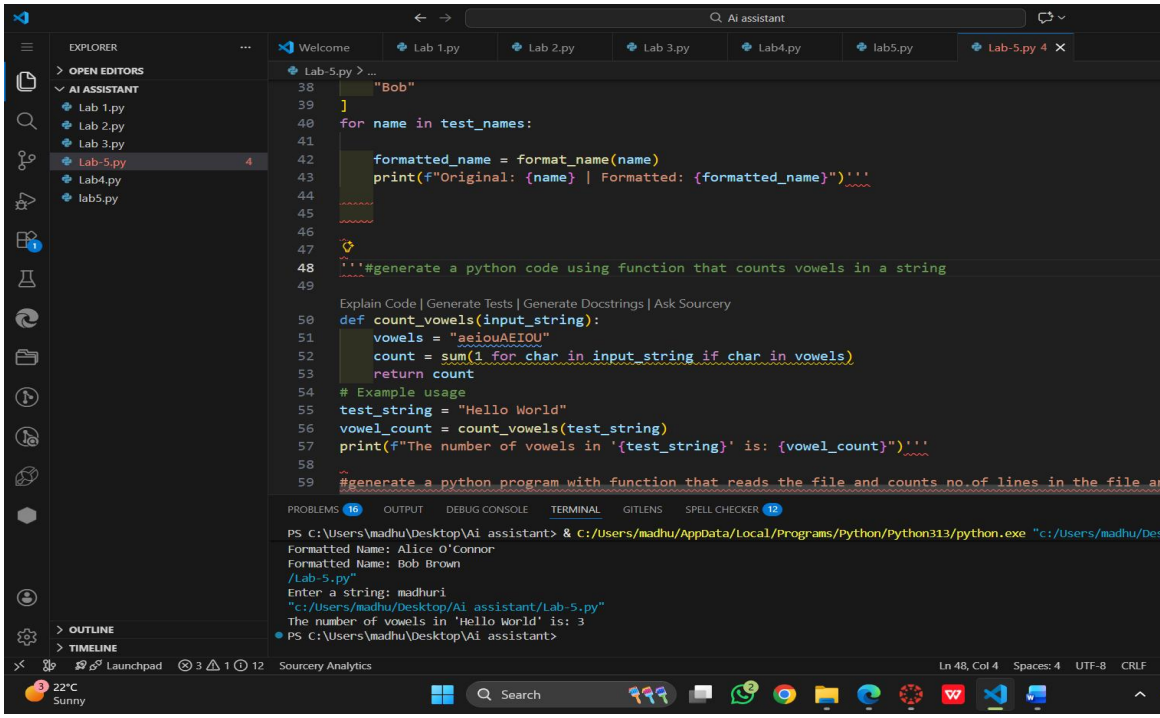
PROMPT:
generate a python code using function that counts vowels in a string

CODE:

```
def count_vowels(input_string):
    vowels = "aeiouAEIOU"
    count = sum(1 for char in input_string if char in vowels)
```

```
    return count
# Example usage
test_string = "Hello World"
vowel_count = count_vowels(test_string)
print(f"The number of vowels in '{test_string}' is: {vowel_count}")
```

OUTPUT:



EXPLANATION:

- The function counts vowels in a given string using a direct logic approach.
- Zero-shot prompting applies the logic without examples.
- Few-shot prompting helps by showing patterns before execution.
- The function returns the total number of vowels in the input string.

TASK-5:

PROMPT:

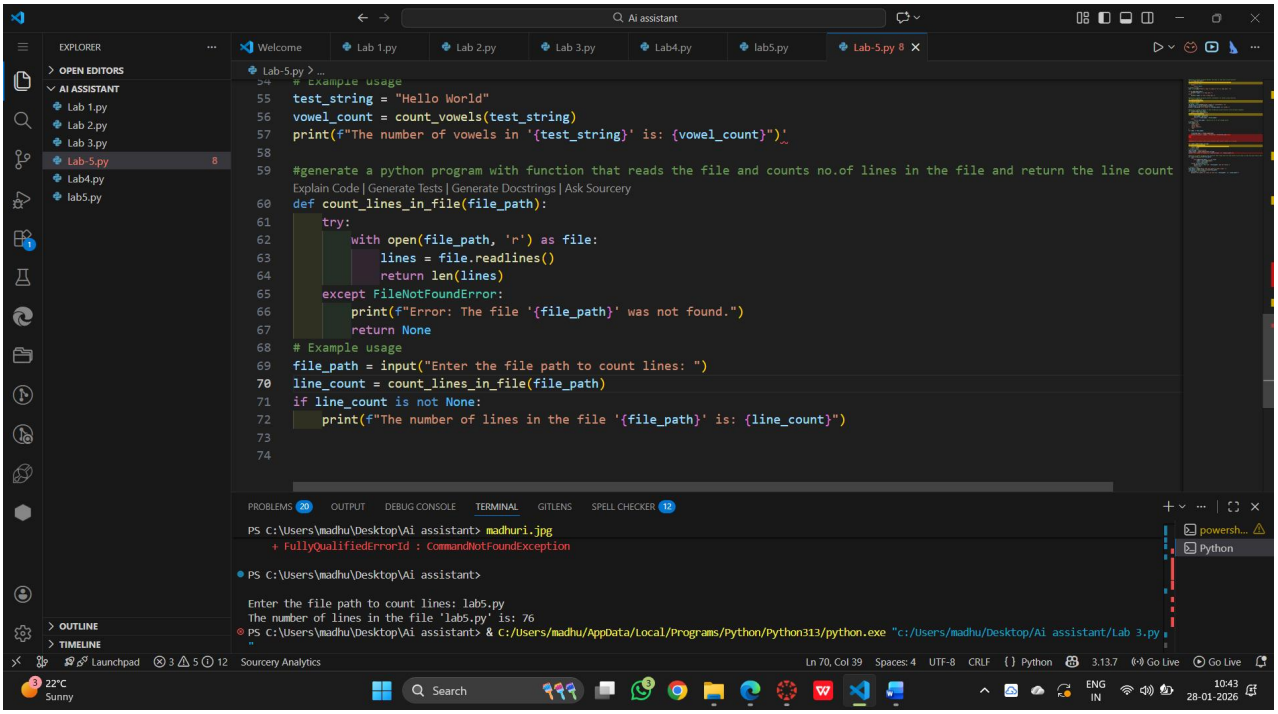
generate a python program with function that reads the file and counts no.of lines in the file and return the line count

CODE:

```
def count_lines_in_file(file_path):
    try:
        with open(file_path, 'r') as file:
            lines = file.readlines()
            return len(lines)
    except FileNotFoundError:
        print(f"Error: The file '{file_path}' was not found.")
        return None
# Example usage
file_path = input("Enter the file path to count lines: ")
line_count = count_lines_in_file(file_path)
```

```
if line_count is not None:
    print(f"The number of lines in the file '{file_path}' is: {line_count}")
```

OUTPUT:



EXPLANATION:

- This program reads a text file and counts the number of lines.
- A function opens the file safely and calculates the line count.
- Error handling is used if the file does not exist.
- The final line count is returned and displayed.